Music Recommendation System Using Python and Django

A MINI PROJECT REPORT SUBMITTED BY

Ananya S A	Akhila U Baliga
4NM18CS019	4NM18CS012
VI Semester, A Section	VI Semester, A Section

UNDER THE GUIDANCE OF

Mrs. Divya Jennifer D'souza

Assistant professor GD I

Department of Computer Science and Engineering

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE AWARD OF THE DEGREE OF

Bachelor of Engineering in Computer Science & Engineering

from

Visvesvaraya Technological University, Belagavi

N.M.A.M. INSTITUTE OF TECHNOLOGY

(An Autonomous Institution under VTU, Belgaum)

AICTE approved, (ISO 9001:2015 Certified), Accredited with 'A' Grade by NAAC

NITTE -574 110, Udupi District, KARNATAKA

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

B.E. CSE Program Accredited by NBA, New Delhi from 1-7-2018 to 30-6-2021

May 2021

NITTE EDUCATION TRUST

N.M.A.M. INSTITUTE OF TECHNOLOGY

(An Autonomous Institution affiliated to Visvesvaraya Technological University, Belagavi)

Nitte – 574 110, Karnataka, India

(ISO 9001:2015 Certified), Accredited with 'A' Grade by NAAC

☎: 08258 - 281039 – 281263, Fax: 08258 – 281265

Department of Computer Science and Engineering

B.E. CSE Program Accredited by NBA, New Delhi from 1-7-2018

CERTIFICATE

Certified that the Mini Project work entitled

Music Recommendation System

Is a bonfide work carried out by

Ananya S A : 4NM18CS019

Akhila U Baliga : 4NM18CS012

In partial fulfilment of requirements for the award of Bachelor of Engineering Degree in Computer Science and Engineering prescribed by Visvesvaraya Technology University,

Belgaum during the year 2020-2021

It is certified that all corrections/suggestions indicated for Internal Assessment have been

Incorporated in the report

The mini project report has been approved as it satisfies the academic requirements in respect of the project work prescribed for the Bachelor of Engineering Degree

Name & Signature of Guide(s)	**Name & Signature of HOD**
Mrs. Divya Jennifer D'souza	Dr. Jyothi Shetty
Assistant Profesor, Gd I	Head of the Department
Department of CSE	Department of CSE

ACKNOWLEDGMENT

We believe that our project will be complete only after we thank the people who have contributed to make this project successful.

First and foremost, our sincere thanks to our beloved principal, **Dr. Niranjan N. Chiplinkar** for giving us an opportunity to carry out our project work at our college and providing us with all the needed facilities.

We express our deep sense of gratitude and indebitness to our guide **Mrs. Divya Jennifer Dsouza**, Assistant Professor, Department of Computer Science and Engineering, for her inspiring guidance, constant encouragement, support and suggestion for improvement during the course of our project.

We sincerely thank **Dr. Jyothi Shetty**, Head of Department of Computer Science and Engineering, Nitte Mahalinga Adyantaya Memorial Institute of Technology, Nitte.

We also thank all those who have supported us throughout the entire duration of our project.

Finally, we thank the staff members of the Department of Computer Science and Engineering and all our friends for their honest opinions and suggestions throughout the course of our project.

Ananya S A (4NM18CS019)

Akhila U Baliga (4NM18CS012)

TABLE OF CONTENTS

ABSTRACT

Music is an integral part of our life. We listen to music everyday as per our taste and mood. With the advancement and increase in volume of digital content, the choice for people to listen to diverse type of music has also increased significantly. Thus, the necessity of delivering the most suited music to the listeners has been an interesting field of research in computer science. One of the important measures to deliver the best music to listeners could be his/her personality trait.

Main idea is to develop a recommender system for the task of automatic playlist continuation. Given a set of playlist features, recommender system should be able to generate a list of recommended tracks that can be added to that playlist, thereby 'continuing' the playlist. Spotify has released a dataset called "Million Playlist Dataset" which would be used as a dataset to train our model. Accurate prediction and recommendation of tracks to the respective playlists accomplishes the task of automatic playlist continuation.

The main problem here is to identify the various way on how to recommend next song to the user or the playlist. Whether it should be based on type of music or language or genre or artists or popularity is a very big question to be discussed before working on the recommendation system for music platforms. Better the recommended songs, more likely the user to keep using that music service. Since user cannot search for all songs manually, if the music service provider is able to provide the recommendations of the songs that user would highly like, then there are higher chances of the music service staying relevant and becoming successful.

Since the dataset is very large and has lots of properties, there cannot be a single method which would work as a recommendation system for such large dataset and giving satisfactory results. Therefore, we tested many methods to achieve the goal. Two main methods are user ratings system where the user rated various songs and then the recommended songs were provided to them resulting in highly favourable outputs from the model. But since, you cannot always ask a user to rate the songs but rather songs are liked or disliked in any music service. Therefore, the second method to model the system was based on this idea. The songs were then recommended on basis of various properties of the song liked by the user.

INTRODUCTION

Music recommender systems (MRS) have recently exploded in popularity thanks to music streaming services like Spotify, Pandora and Apple Music. By some accounts, almost half of all current music consumption is by the way of these services. While recommender systems have been around for quite some time and are very well researched, music recommender systems differ from their more common siblings in some characteristically important ways: the duration of the items is less (3-5 min for a song vs 90 minutes for a movie or months/years for a book or shopping item), the size of the catalog of items is larger (10s of millions of songs), the items are consumed in sequence with multiple items consumed in a session, repeated recommendations have a different significance. Music Recommender Systems then require different approaches from traditional recommender systems.

This project's goal is to provide automatic playlist continuation which would enable any music platform (here Spotify) to seamlessly support their users in creating and expanding the playlists by making recommendations based on their choices and preferences. Furthermore, the recommender system does not require any rich and varied supply of user data, instead requiring only basic information as input such as the title of the playlist, the tracks currently in the playlist, and the artists associated with those tracks.

First, we need to determine the business objectives.

i) Objective of the project is to create a recommender system for automatic playlist continuation.

ii) Given a set of playlist features, recommender systems should be able to generate a list of recommended tracks that can be added to that playlist, thereby 'continuing' the playlist.

Then we need to decide on the plan to approach this dataset to train our model. Processing the dataset and finding correlations between the various tracks provided. Using various classification techniques, generating the next song for the playlist which is based on the preferences of the user.

There may be different approaches such as:

Non-Personalized Approaches:

It includes recommending which is popular across the whole system. Simple and relatively easier to implement.

Collaborative Filtering:

Personalized recommendations, based on similarity between either users or items. User-based CF assumes that users who behaved similarly (buying patterns, preference in movies or music, job application) in the past will behave similarly again in the future.

Content Based Filtering:

The actual content of item refers to the actual characteristics of an item and these attributes are broken down into 'tags' and items are then matched to user preferences (as modelled by the tags) accordingly.

SOFTWARE REQUIREMENTS

Python:

Python is a programming language that distinguishes itself from other programming languages by its flexibility, simplicity, and reliable tools required to create modern software.

Python is consistent and is anchored on simplicity, which makes it most appropriate for machine learning.

Python is a programming language that is preferred for programming due to its vast features, applicability, and simplicity. The Python programming language best fits machine learning due to its independent platform and its popularity in the programming community

Django:

Django is a high-level Python Web Development framework that encourages rapid development and clean, pragmatic design. It has been built by experienced developers, and takes care of much of the hassle of Web development. It is also free and open source.

Django REST Framework is a powerful and flexible toolkit for building Web APIs which can be used to Machine Learning model deployment. With the help of Django REST framework, complex machine learning models can be easily used just by calling an API endpoint.

HTML:

First developed by Tim-Berners-Lee in 1990, **HTML** is short for **Hypertext Markup Language**. HTML is used to create electronic documents (called pages) that are displayed on the World Wide Web. Each page contains a series of connections to other pages called hyperlinks. Every web page you see on the Internet is written using one version of HTML code or another.

HTML code ensures the proper formatting of text and images for your Internet browser. Without HTML, a browser would not know how to display text as elements or load images or other elements. HTML also provides a basic structure of the page, upon which Cascading Style Sheets are overlaid to change its appearance. One could think of HTML as the bones (structure) of a web page, and CSS as its skin (appearance).

HARDWARE REQUIREMENTS

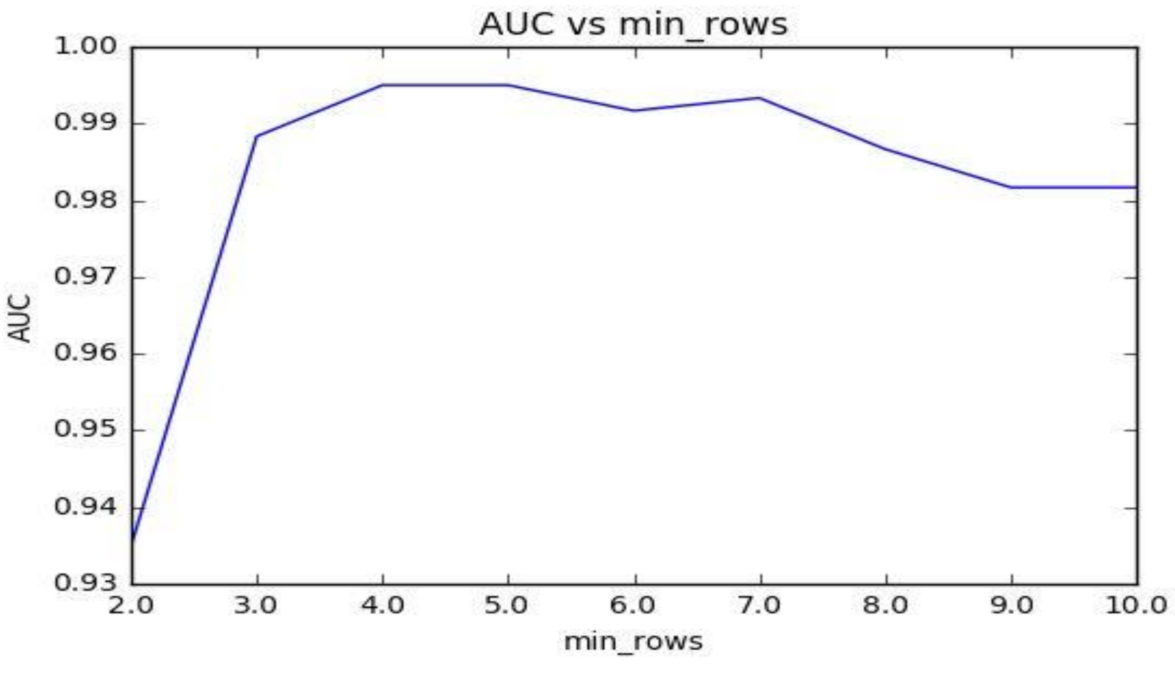
- **RAM** : Minimum 4GB (8GB recommended)
- **Processor** : Intel i3 or above
- **HDD** : 20 GB Disk space

IMPLEMENTATION

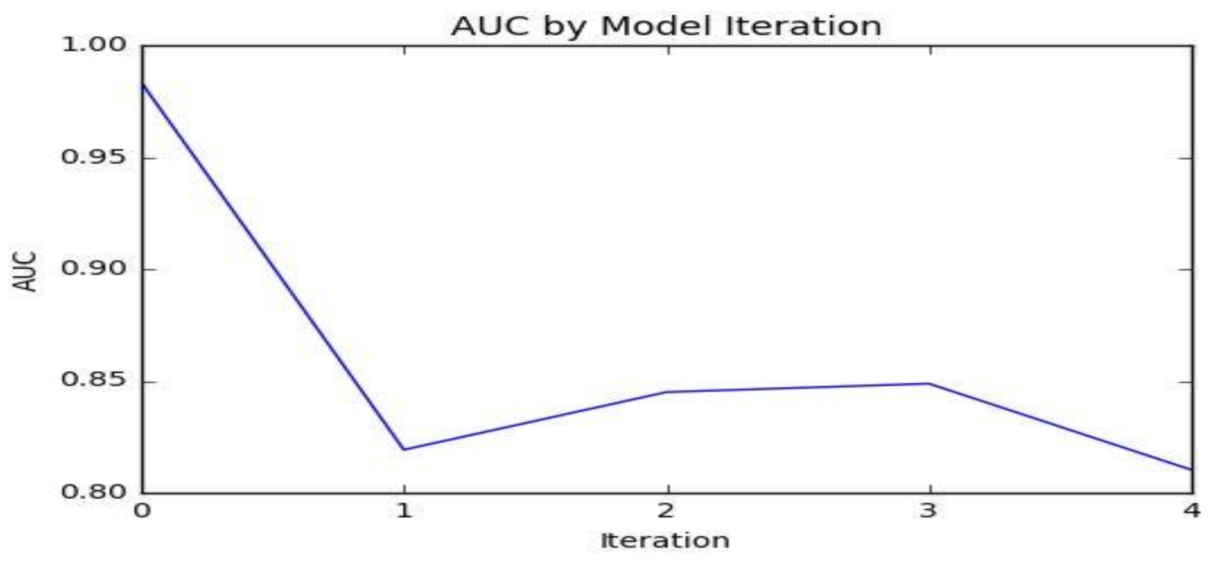
A recommendation system was built on a small training dataset, though the random forest algorithm should alleviate this to a certain extent through the process of bagging on both sample and predictor selection. To further mitigate this, we can tune parameters min_rows and ntrees, where min_rows specifies the minimum number of observations for a leaf in order to split and ntrees specifies the number of trees to build. Ideally we would also tune the maximum depth to which a tree could be built, but during testing we found that setting a low maximum depth would result in too many short trees with few leaf nodes due to the small training dataset. This would cause "like" probabilities to be bucketed into several discrete values and prevent a definitive ranking from which we would determine the top 10 songs. By allowing trees to grow to their full extent (i.e. no maximum depth), we can obtain granularized probabilities for ranking without sacrificing significant predictive power.

We can select the right combination of min_rows and ntrees through cross-validation, employing the "leave-one-out" method and averaging AUC over all of the resultant models for a given combination of parameters. For the purpose of tuning, I had my roommate complete the initial stage of the program, rating 10 random songs to create a training dataset. I used this training data to build 90 different models using different combinations of min_rows and ntrees, then grouped by min_rows and ntrees and averaged AUC values.

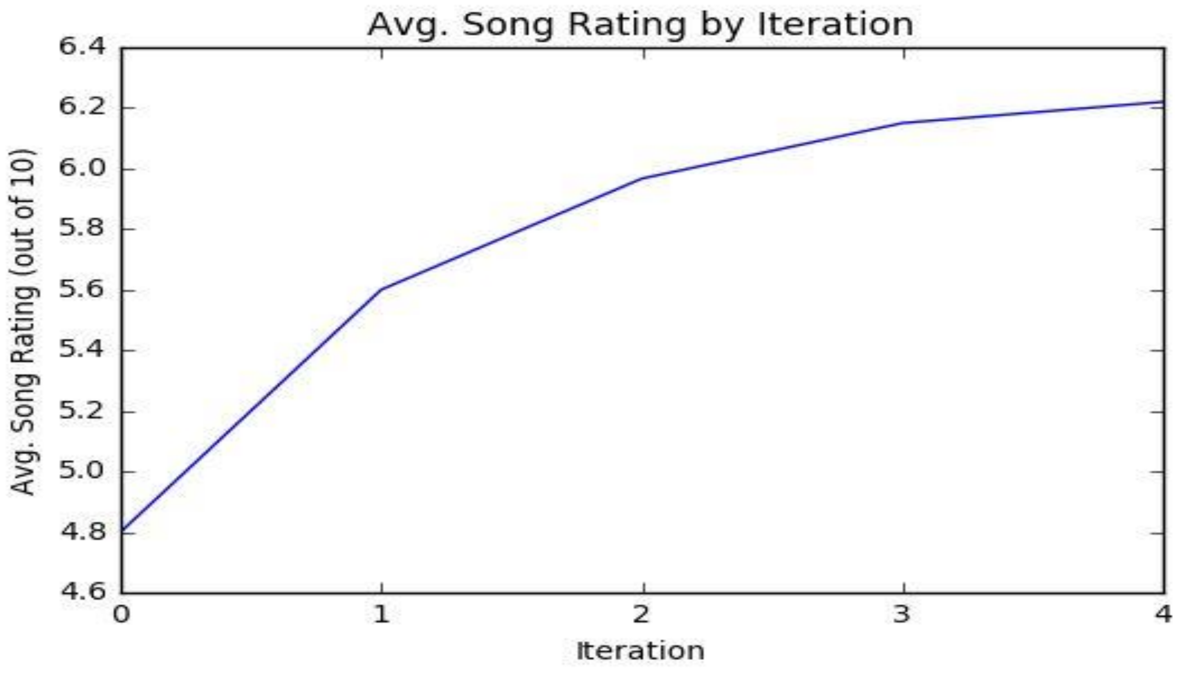
Here we see that AUC stays fairly consistent for min_rows >2



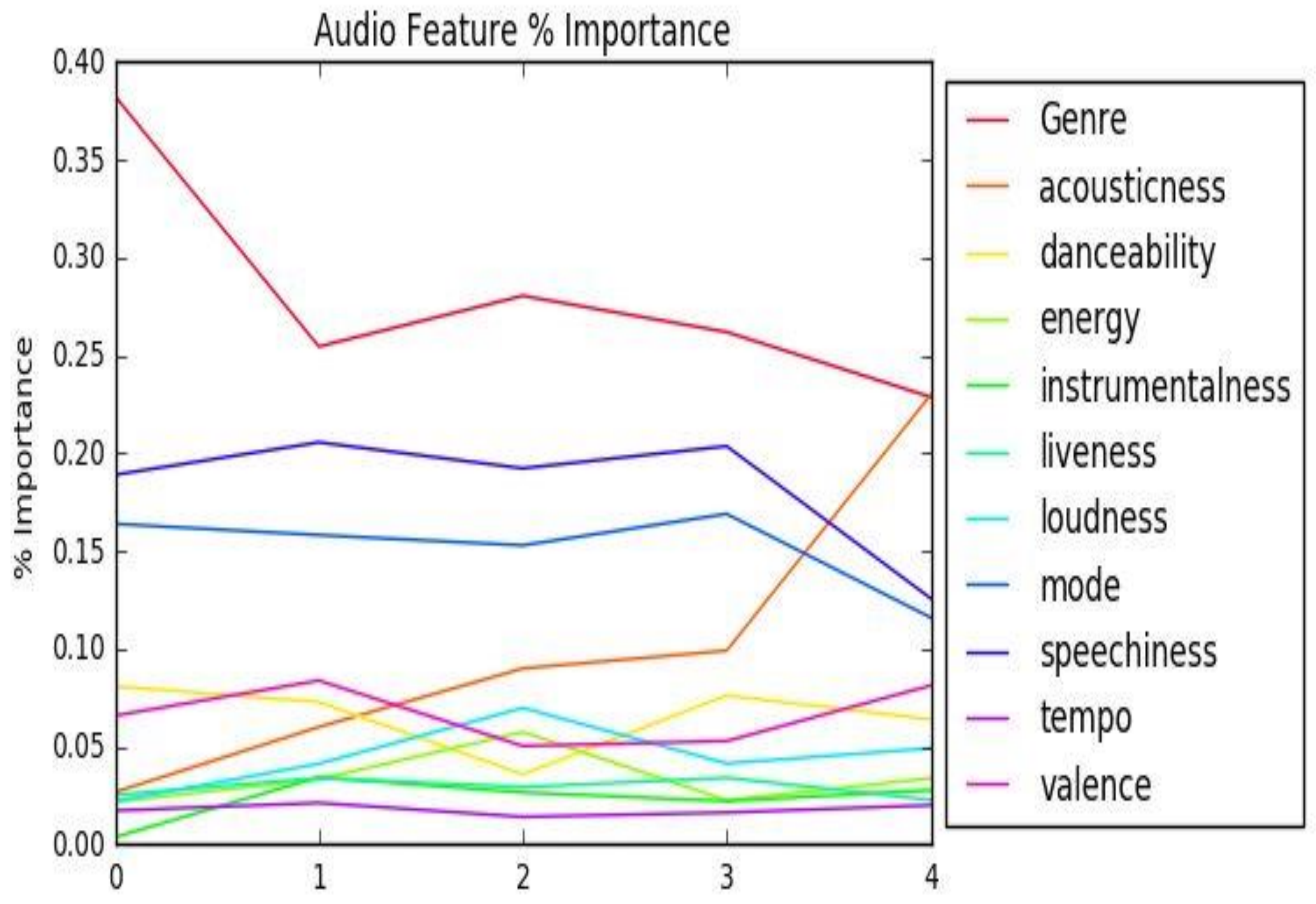
Once we are satisfied with the tuning of the model, we can begin actually making predictions. Rating the top 10 songs recommended by each iteration of the model. We can observe how the model improves over each iteration through a few different metrics.



Averaging ratings for each set of recommended songs, the plot in Figure 6 indicates that on average, the user's ratings increase after each iteration of the model.



We can also utilize the model's ranking of variable importance to draw some conclusions about the user's musical tastes. This user ranks music primarily by genre, but also considers speechiness (a measure of the presence of spoken words on a track), mode (either major or minor scale), and acousticness



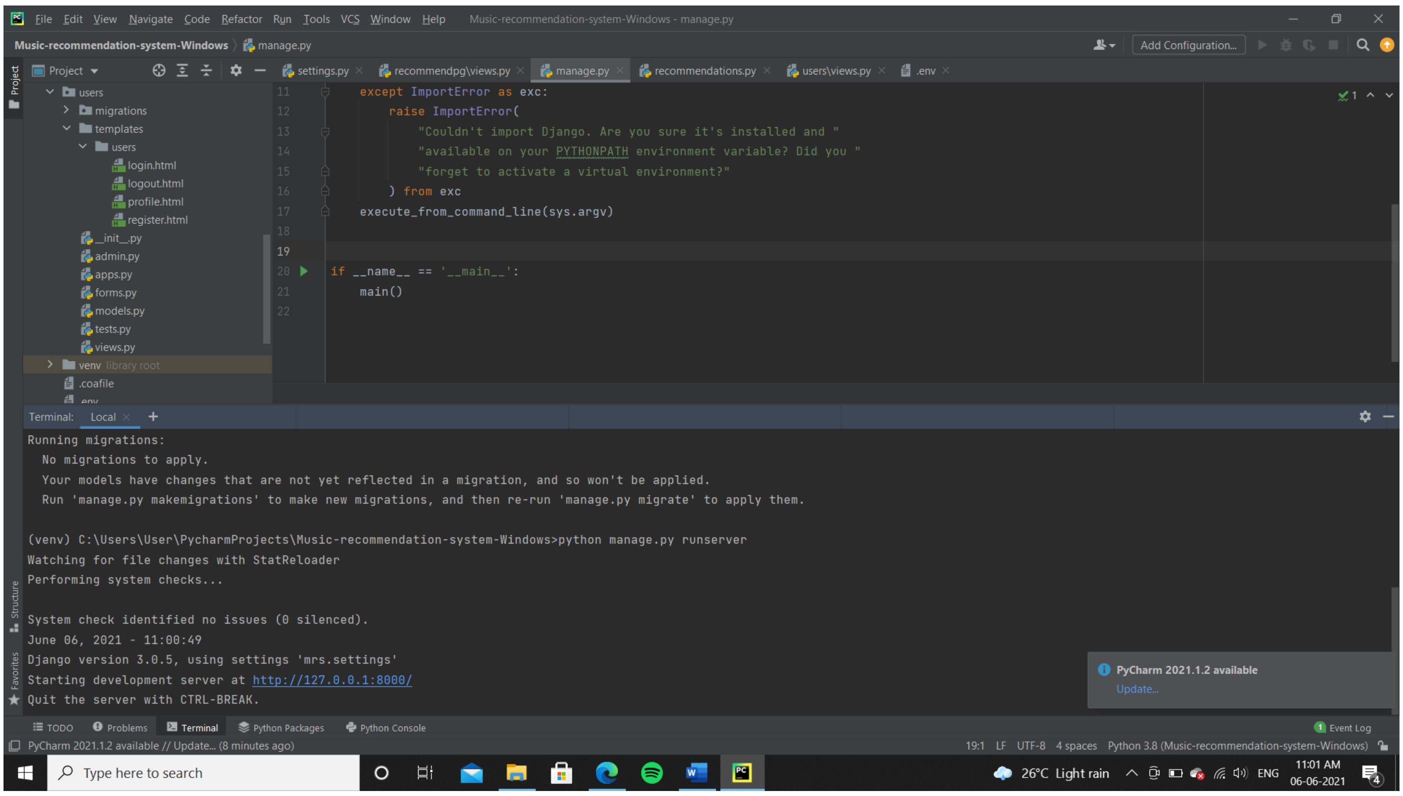
Training Models used: Essemble Random Forest

```
from sklearn.ensemble import RandomForestClassifier
import pandas as pd
import platform
import math

def recommendations(sp,tid):
    features = pd.DataFrame(sp.audio_features(tid))
    if (platform.system() == 'Windows'):
        f = open('E:\Desktop\ml project\songDB.tsv', 'r')

    else:
        f = open('/home/arnamaity/Hackathons/Music-recommendation-system/spotify-music-
genre-list/songDb.tsv', 'r',
             encoding="ISO-8859-1")

    y = f.read().splitlines()
    count = []
    for i in range(len(y)):
        if (i % 2 == 0):
            count.append(y[i].split('\t'))

    d = []
    e = 0
    f = []
    for i in range(len(count)):
        d.append(count[i][-1])

    i = 1
    j = i + 1
```

```
# MODEL

p = []
model = RandomForestClassifier(n_estimators=19, max_depth=2)
X_train = [count[i][1:11] for i in range(1, len(count))][:92106]
y_train = g[:92106]
X_test = features.iloc[:, 1:11]
model.fit(X_train, y_train)
p.append(model.predict(X_test))
print(p)
p[0].sort()

# print('GETTING YOUR RECOMMENDATION....')
```

```
# DICTIONARY
dictionary = dict(zip(g, d))
# print(dictionary)
# print(dictionary[fa(p[0])])

'-----------------------------------------------------------------------------------------------------------------------------------------------------------------------
_'
# RETRIEVING THE NAMES BASED ON GENRE AND DURATION IN MS
i = 0
seu = []
try:
    for j in f:
        if (dictionary[fa(p[0])] == count[i][-1]):
            while (i != j + 1):
                v = float(count[i][-3])
                seu.append(v)
                i += 1
            break
        else:
            i += j
except IndexError:
    pass

seu.sort()
recommendations = []

try:

    for i in range(1, len(count)):
        if (str(seu[-i]) == count[i][-3]):
            tmp = dict()
            t = sp.track(count[i][14].split(':')[2])
            tmp['id'] = t['id']
            tmp['name'] = t['name']
            tmp['art'] = t['album']['images'][1]['url']
            tmp['link'] = t['external_urls']['spotify']
            tmp['album'] = t['album']
            tmp['duration'] = str(math.floor(t['duration_ms']/60000)) + " mins " +
str(math.floor((t['duration_ms']/60000 - math.floor(t['duration_ms']/60000))*60)) + "
secs "
            recommendations.append(tmp)
except IndexError:
    pass

context = {
    'tracks': recommendations
}
return context
```

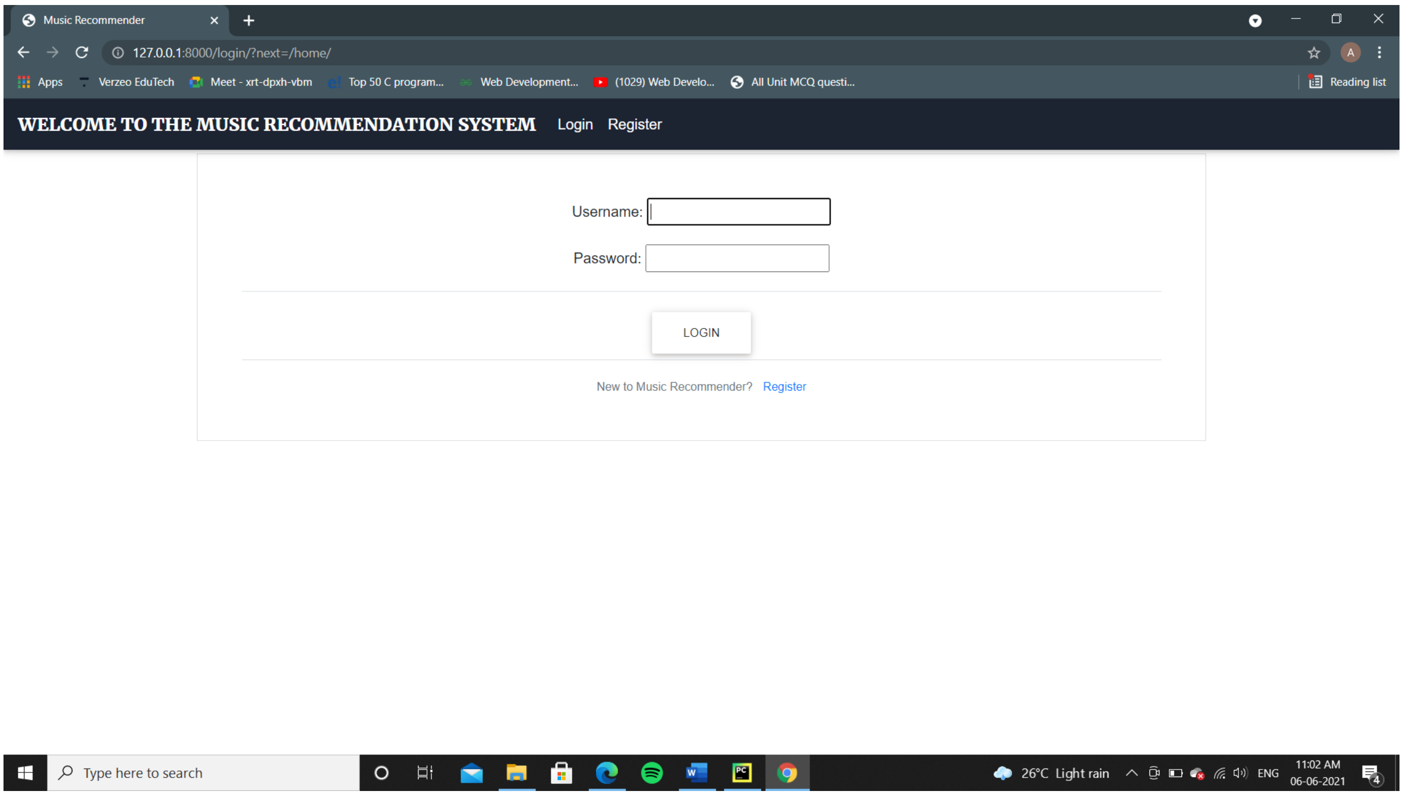
RESULT


File Edit View Navigate Code Refactor Run Tools VCS Window Help Music-recommendation-system-Windows - manage.py
Music-recommendation-system-Windows › manage.py
Add Configuration...
Project
settings.py
recommendpg\views.py
manage.py
recommendations.py
users\views.py
.env
users
migrations
templates
users
login.html
logout.html
profile.html
register.html
__init__.py
admin.py
apps.py
forms.py
models.py
tests.py
views.py
venv library root
.coafile
Terminal: Local
PyCharm 2021.1.2 available
Update...
TODO Problems Terminal Python Packages Python Console Event Log
PyCharm 2021.1.2 available // Update... (8 minutes ago)
19:1 LF UTF-8 4 spaces Python 3.8 (Music-recommendation-system-Windows)
Type here to search
26°C Light rain
ENG
11:01 AM
06-06-2021

```
    except ImportError as exc:
        raise ImportError(
            "Couldn't import Django. Are you sure it's installed and "
            "available on your PYTHONPATH environment variable? Did you "
            "forget to activate a virtual environment?"
        ) from exc
    execute_from_command_line(sys.argv)


if __name__ == '__main__':
    main()
```

```
Running migrations:
  No migrations to apply.
  Your models have changes that are not yet reflected in a migration, and so won't be applied.
  Run 'manage.py makemigrations' to make new migrations, and then re-run 'manage.py migrate' to apply them.

(venv) C:\Users\User\PycharmProjects\Music-recommendation-system-Windows>python manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).
June 06, 2021 - 11:00:49
Django version 3.0.5, using settings 'mrs.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
```

Login Page



WELCOME TO THE MUSIC RECOMMENDATION SYSTEM Login Register

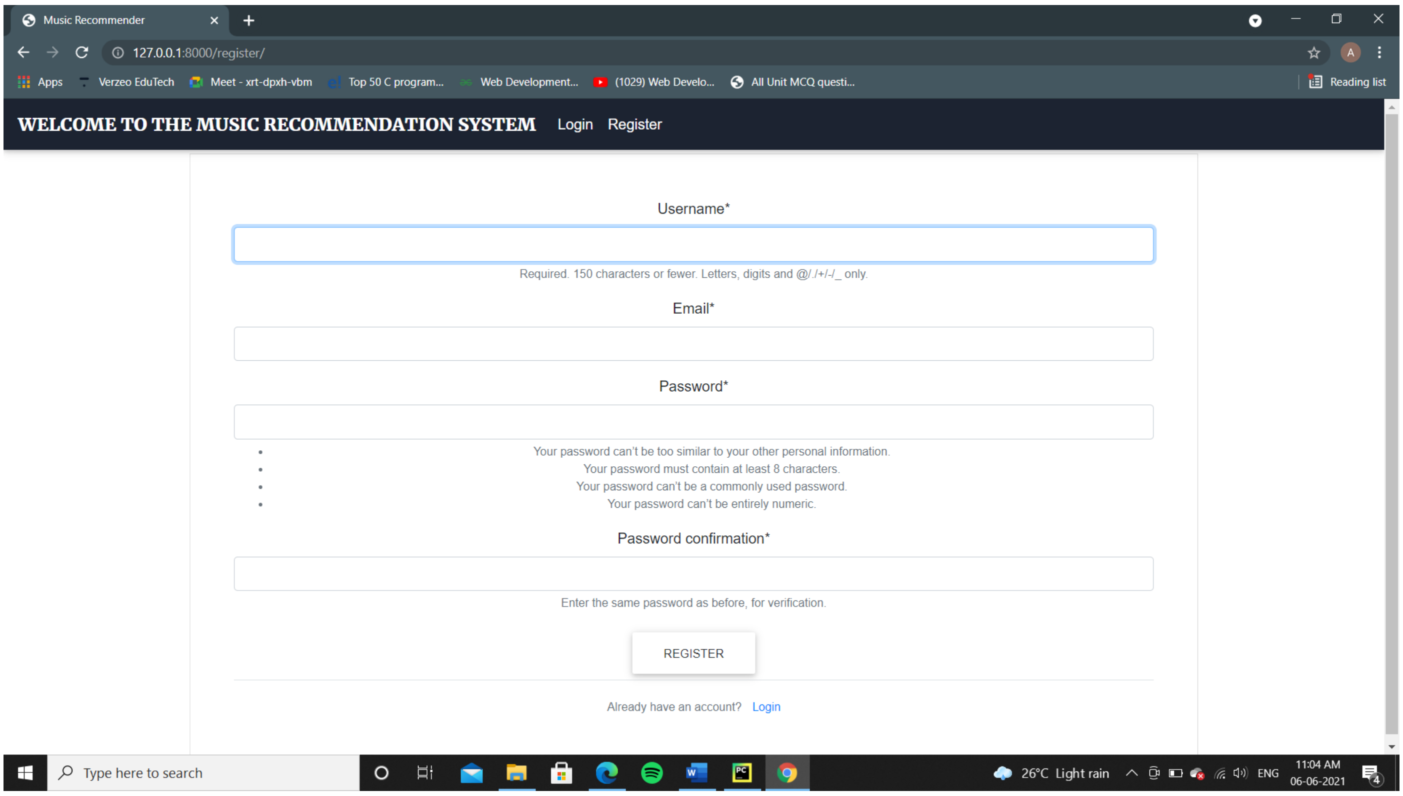
Username:

Password:

LOGIN

New to Music Recommender? Register

Register Page


127.0.0.1:8000/register/

WELCOME TO THE MUSIC RECOMMENDATION SYSTEM Login Register

Username*

Required. 150 characters or fewer. Letters, digits and @/./+/-/_ only.

Email*

Password*

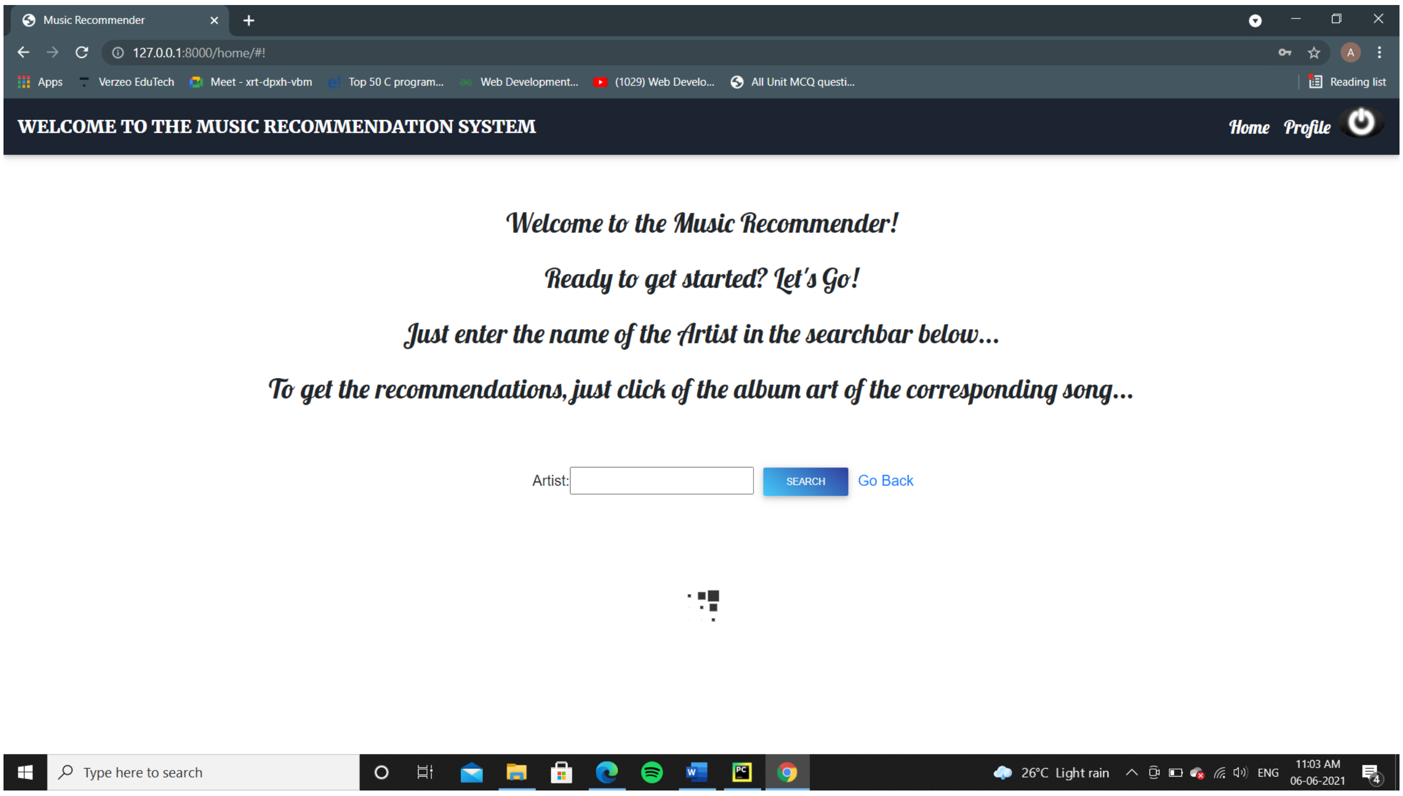
- Your password can't be too similar to your other personal information.
- Your password must contain at least 8 characters.
- Your password can't be a commonly used password.
- Your password can't be entirely numeric.

Password confirmation*

Enter the same password as before, for verification.

REGISTER

Already have an account? Login


127.0.0.1:8000/home/#!

WELCOME TO THE MUSIC RECOMMENDATION SYSTEM *Home* *Profile*

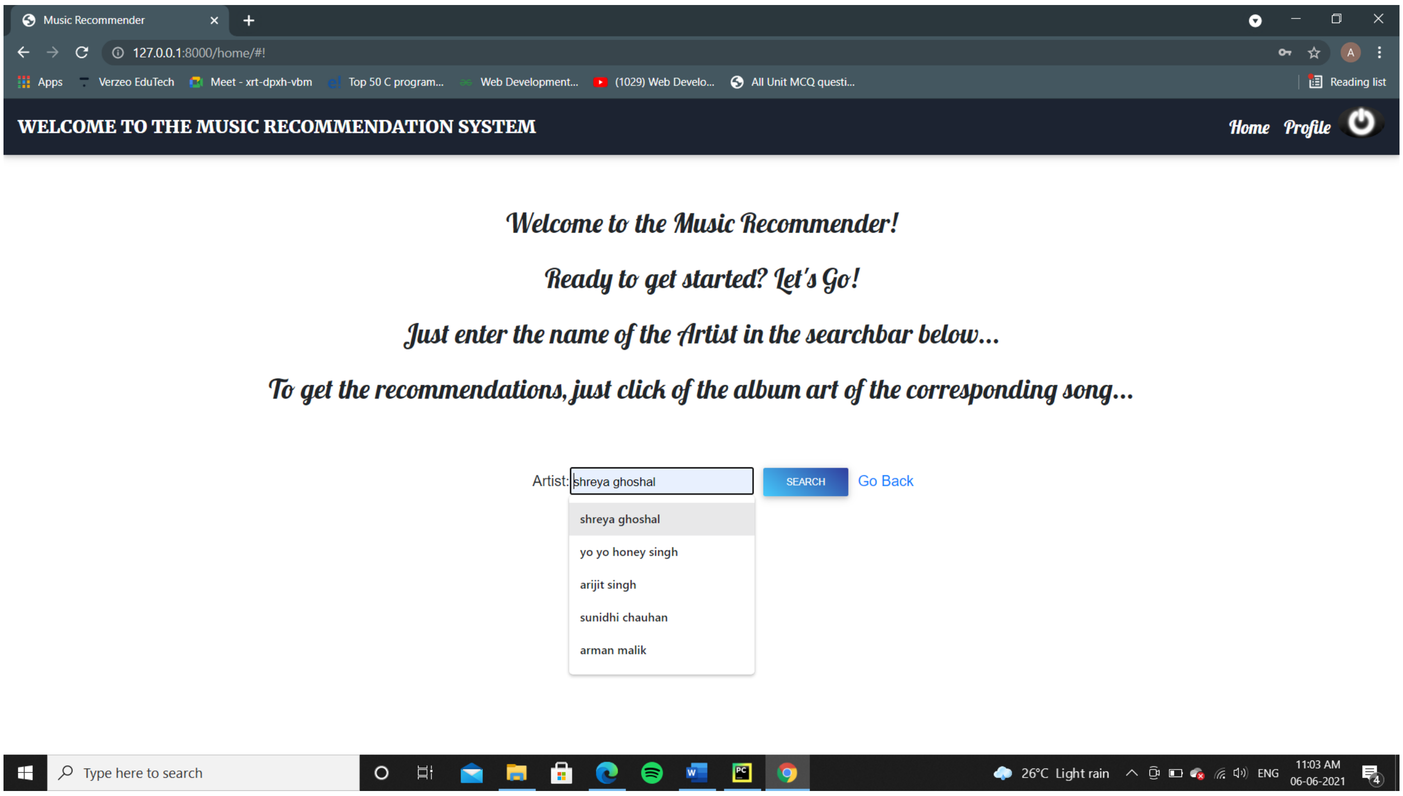
Welcome to the Music Recommender!

Ready to get started? Let's Go!

Just enter the name of the Artist in the searchbar below...

To get the recommendations, just click of the album art of the corresponding song...

Artist: SEARCH Go Back



WELCOME TO THE MUSIC RECOMMENDATION SYSTEM

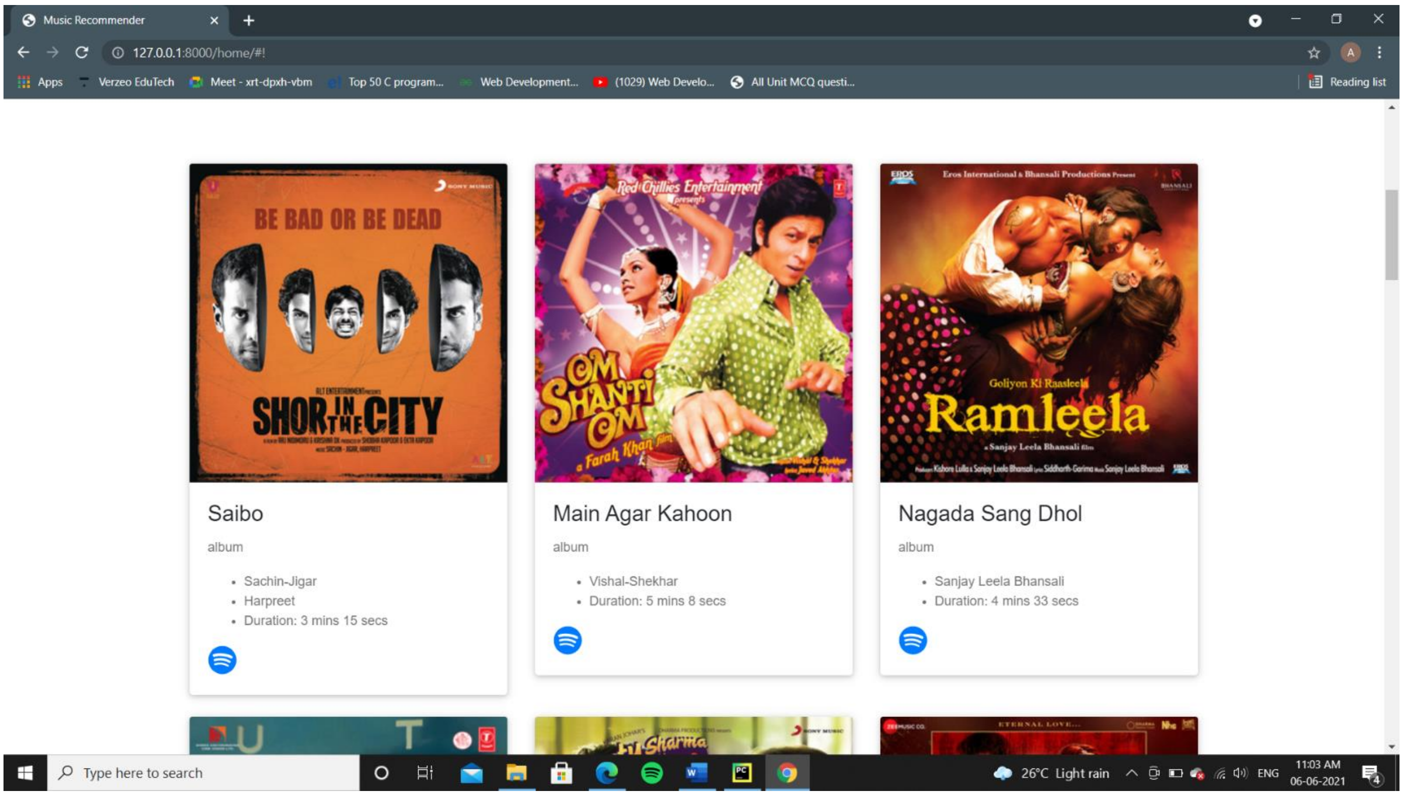
Welcome to the Music Recommender!

Ready to get started? Let's Go!

Just enter the name of the Artist in the searchbar below...

To get the recommendations, just click of the album art of the corresponding song...

Artist: shreya ghoshal



Saibo

album

- Sachin-Jigar
- Harpreet
- Duration: 3 mins 15 secs

Main Agar Kahoon

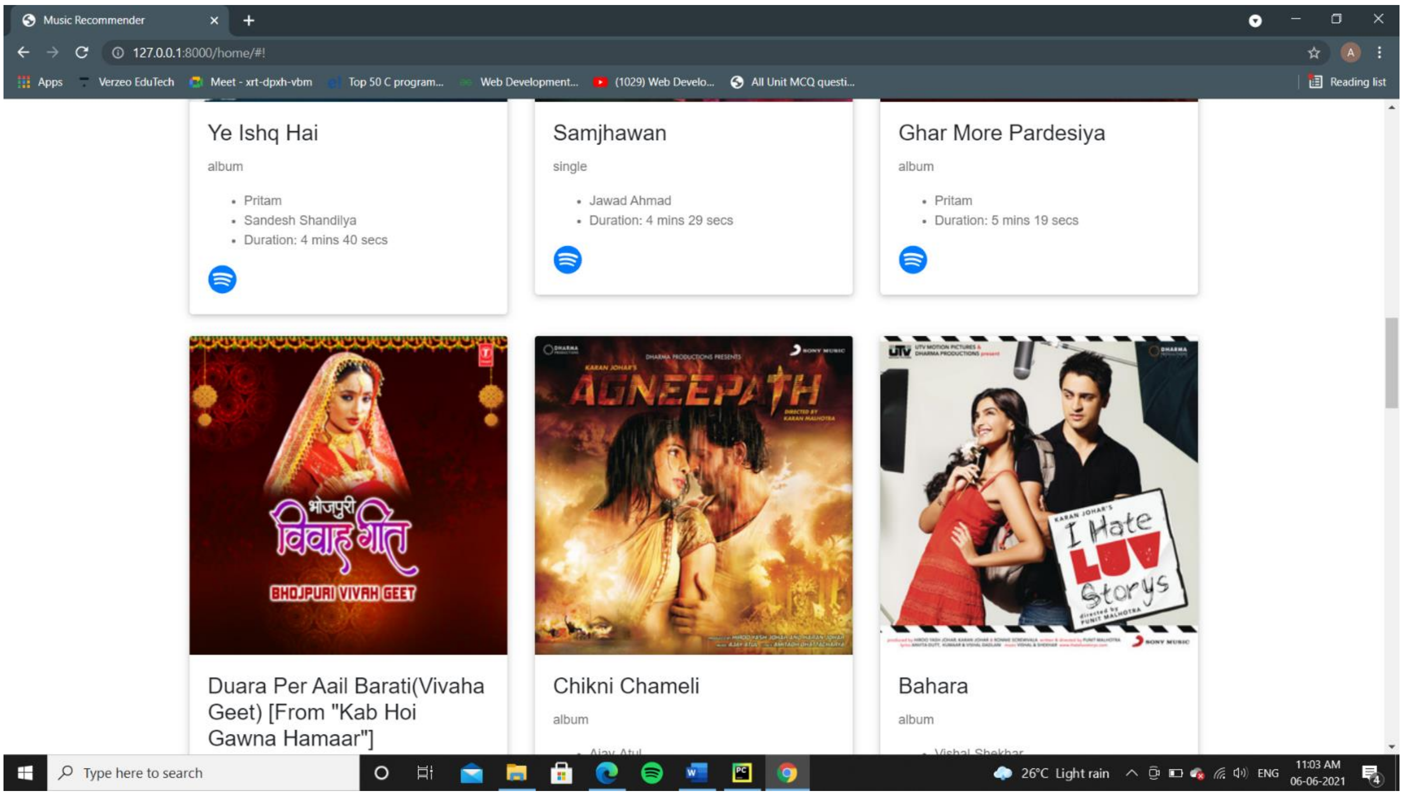
album

- Vishal-Shekhar
- Duration: 5 mins 8 secs

Nagada Sang Dhol

album

- Sanjay Leela Bhansali
- Duration: 4 mins 33 secs



Music Recommender

127.0.0.1:8000/home/#!

Ye Ishq Hai
album
- Pritam
- Sandesh Shandilya
- Duration: 4 mins 40 secs

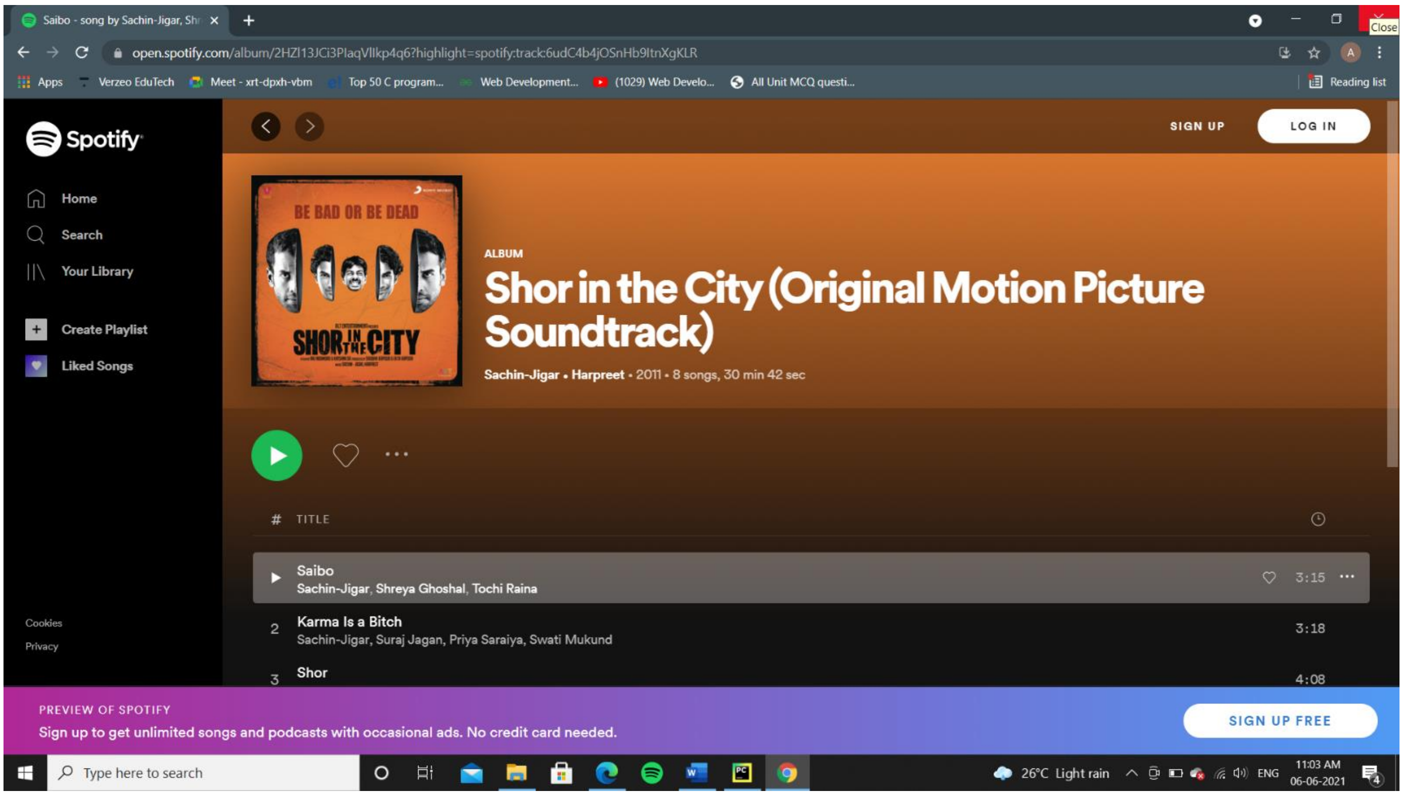
Samjhawan
single
- Jawad Ahmad
- Duration: 4 mins 29 secs

Ghar More Pardesiya
album
- Pritam
- Duration: 5 mins 19 secs

Duara Per Aail Barati(Vivaha Geet) [From "Kab Hoi Gawna Hamaar"]

Chikni Chameli
album
- Ajay-Atul

Bahara
album
- Vishal-Shekhar



Saibo - song by Sachin-Jigar, Shr

open.spotify.com/album/2HZI13JCi3PlaqVllkp4q6?highlight=spotify:track:6udC4b4jOSnHb9ltnXgKLR

ALBUM

Shor in the City (Original Motion Picture Soundtrack)

Sachin-Jigar • Harpreet • 2011 • 8 songs, 30 min 42 sec

#	TITLE	
	Saibo Sachin-Jigar, Shreya Ghoshal, Tochi Raina	3:15
2	Karma Is a Bitch Sachin-Jigar, Suraj Jagan, Priya Saraiya, Swati Mukund	3:18
3	Shor	4:08

PREVIEW OF SPOTIFY
Sign up to get unlimited songs and podcasts with occasional ads. No credit card needed.

SIGN UP FREE

CONCLUSION

Recommendation systems are very prevalent nowadays and can be seen playing in the background of most websites and apps we visit. Whether we are visiting an e-commerce website like Amazon, a travel website such as Expedia, entertainment apps like Netflix, YouTube and Spotify, recommendation systems are an inevitable aspect. The inevitability arises due to the need to stay more relevant in business, acquire more customers and deliver an absolutely fabulous customer experience. In our project, we describe and attempt at developing one such recommendation system.

References

1.Spotify Data set

https://www.kaggle.com/yamaerenay/spotify-dataset-19212020-160k-tracks

2. GeeksforGeeks

https://www.geeksforgeeks.org/